Recommended Practices for AP-238 CC1

David Odendahl, Boeing/Tulsa, david.j.odendahl@boeing.com
Martin Hardwick, STEP Tools Inc., hardwick@steptools.com

3 January 2005

Purpose

This document is not intended to be a complete tutorial on how to develop a writer for AP238 CC1. Rather, it is a collection of examples, advice and warnings on how to create such a tool. I expect that the information contained here will greatly reduce the time required for a programmer to create similar software.

This is free advice, so try not to get too mad at me if something in here isn't correct or unclear. However, please let me know what you find incorrect or confusing, and I'll try to get it fixed. Thanks!

Audience

This document assumes that the reader is an experienced programmer who understands the Object Oriented paradigm. It also assumes that the reader understands the basic concepts of CNC, including CL file formats, machine dependencies, and RS-274.

Background

In March, of 2004, I embarked on an effort to create a tool that would allow the conversion of a CATIA V5 CL file into an AP238 CC1 Part 21 file. I also created tools to convert from AP238 into three separate CNC formats: Siemens 840D, Fanuc 15/16/30, and ACL (RS-494). The purpose of this was to demonstrate the practicality of the AP238 data format, and provide data that would allow the machining of test parts using AP238. After several false starts, and about 300 hours of effort, I competed all four converters in August of 2004. The data produced using these converters have been used to manufacture test parts that demonstrate the benefits of AP238. This document conforms to the AP238 FDIS version, which was finalized in October of 2004.

The Task at Hand

The creation of a writer for AP238 shouldn't be TOO big of a job. The data presented in a CL file can be translated on a mostly one-to-one relationship with an AP238 data structure. CC1 is a very small subset of the AP238 standard. It's easier to create an AP238 writer, since all possible permutations of AP238 CC1 need be used. A reader, however, has a tougher job, since it must support ALL of the AP238 CC1 representations.

Thanks

Thanks are due to Mauro Costa of Boeing, and Dave Loffredo, of Step Tools for reviewing this document. Xun Xu, of the University of Auckland also provided valued feedback.

The Challenges

There are four main challenges to the creation of the software:

- The "**Part Twenty-What**? Problem" : How does one understand what a Part 21 file is, and specifically, how does one extract the relevant information from the gazillion pages of ISO 10303/14649 documentation so that one can write or read such a file.
- The "**Which Spec, Now**?" : How does one locate and understand the definition of AP238, which is also contained in a gazillion pages of various STEP standards documents.
- The "**What's a Mapping Table, Anyhow**" Problem : How does one understand the hundreds of pages of mapping tables in the AP238 specification?
- The "**The Wild, Wild West**" Problem : How does one adapt the present unstructured, "anything goes" format of NC data to the object-oriented, highly structure of AP238?

Part Twenty-What?

AP238 files are represented in a "Part 21" file format. The term "Part 21" refers to the fact that the file format is defined using ISO standard 10303-**21**. A Part 21 file is a list of ASCII data, referred to as "entities". These entities contain attributes, which can be a numerical id, data, comments, and pointers to other entities.

Get a Part 21 Browser

An AP238 file is likely to contain thousands of entities. Because of the universal, non-redundant nature of STEP data, these pointers are likely to be pointing in an awkward direction...that is, the pointers are just as likely to be pointing up the data hierarchy as to be pointing down. This makes it hard to read or write this data. The ↑↓ arrows in the example programs show how the entities point to each other in a Part 21 file. A careful inspection of these example files should illustrate this difficulty.

With all these different entities pointing in all these different directions, it's difficult to figure out what's happening in an AP238 Part 21 file. I used a tool, called "STEP File Browser" from STEP Tools, Inc. to help me understand the files I generated and read. The browser allows you to quickly follow pointers, (including the ones pointing the "wrong" way) as well as quickly view the name, type and value of the attributes in each entity.

It's pretty tough to develop software for a file that's hard to read. Get an appropriate tool to allow you to see what you've done. You will save a lot of time.

Label Your Entities

Most (all?) entities contain an optional label attribute. Label each entity with relevant data that identifies the context of the entity. This will be extremely helpful when understanding and debugging a Part 21 file. Make sure that you provide for a simple way to eliminate the optional entity labels once you've completed debugging, as these are likely to significantly improve file size.

Be careful, though… in many cases the contents of the labels (like the name on representation_items) is actually specified by the AP. In this case, a misplaced comment will cause your program to fail!

An Example Entity

```
#4727= MACHINING_TOOLPATH_SEQUENCE_RELATIONSHIP ('',$,#4973,#4229,0.);
  ↑           ↑                                  ↑ ↑   ↑     ↑   ↑
                                                                  The sequence of the relationship
                                                            The ID of the related method
                                                       The ID of the related method
                                                  An optional description of the entity
                                              The entity's name
              The entity's type
 The entity's ID, which can be used as a reference by other entities
```

Which Spec, Now?

One of the most powerful aspects of the STEP suite of standards is the fact that the various APs and standards are harmonized with each other. AP238 is not so much a new standard, but the careful arrangement and use of data structures that have already been established in other STEP standards.

While this makes the standard extremely powerful, it makes it a bit rough to work with the standard, especially when one is new to it. It's easy to lose track of where you are when you are following the links from standard to standard. Below is some basic information that describes the relevant documents. One should have these documents at hand when developing software for STEP-NC. (Hope you have a big desk, or are able to view them electronically!… These are BIG documents!)

I'd start by first reading 14649-1.

ISO/DIS 10303 – 238:

The "AP238"document. Contains definitions, mapping tables, information requirements, conformance classes, and EXPRESS listings. You'll spend most of the time in this document .

ISO 14649-1
An overview of the STEP-NC concept. Contains very good scenarios in the annexes which describe the benefits and possible implementation of STEP-NC

ISO 14649-10
Contains basic definitions and structures. Describes workingstep and toolpath.

ISO 14649-11
Contains definitions and structures specifically for milling applications.

ISO 14649-111
Contains definitions and structures for machining cutting tools

ISO 10303-11 Annex E
Contains a description of the EXPRESS-
G representation. This is useful when trying to understand the ARM

ISO 10303-42
Contains definitions and structures for basic geometric entities: polyline, etc.

What's a Mapping Table, Anyhow?

A key component of the AP238 specification is mapping tables. Mapping tables define the relationship between various entities and how they must be structured in a Part 21 file. An explanation of the notation for the mapping tables is located at the front of the mapping tables in the AP238 document. In the examples, the relevant mapping tables are displayed below the entity Ids. An explanation of many of the entities is also contained in the examples. Have a look at them. The comments help with the understanding of how the mapping tables work (I hope). I found it easier to have a hard copy of the mapping table notation in front of me when I am wading through the tables.

The Wild, Wild West

For the most part, in a traditional NC data file, pretty much anything goes. Very few restrictions are placed on the order or structure of the data that is sent to the CNC. Each instruction is simply executed in the order in which it is received. I've seen lots of weird stuff in these files…. I've

seen spindles turned on twice before it is turned off, feedrates selected, but never used, and tools selected, but never used. I used the following methodology to go from the Wild, Wild West of a traditional CL file to the highly structured environment of AP238. This methodology isn't guaranteed to work efficiently in every situation, but has worked well for me.

1. Workingsteps are tied to tool changes
    a. Create a new, temporary, empty, workingstep at the beginning of the file
    b. Populate the workingstep with data (tool selection, tool paths, spindle commands, etc..) until a tool change is encountered.
    c. Verify that all required data is present in the temporary workingstep
    d. Output the temporary workingstep as a workingstep in the output file
    e. Create a new, empty, temporary workingstep that contains the new, selected tool.

2. Toolpaths are tied to feedrate changes
    a. Create a new, temporary, empty, toolpath at the beginning of the file. The empty toolpath should contain room for tool motion and orientation
    b. Populate the toolpath with data (polyline coordinates) until a feedrate change is encountered
    c. If no tool orientation change is commanded, disregard this information
    d. Verify that all required data is present in the temporary toolpath
    e. Save the new toolpath and link to the active workingstep
    f. Create a new, empty, temporary toolpath that contains the new feedrate

Other notes

- Commonly used tool orientation vectors can be reused ex: (0.0, 0.0, 1.0)
- If there are a lot of feedrate changes, it may be smarter to represent the feedrate as a polyline within the toolpath, instead of a constant

ARM and AIM

The STEP documentation frequently refers to the ARM and AIM. Simply put, the ARM section presents a higher-level description of the AP238 data structure. It's easiest to read in the EXPRESS-G format. The AIM presents a low-level description of how the ARM is implemented. The relationship between the ARM and AIM is described in the mapping tables.

The Examples

The remainder of this document contains examples of valid AP238 structures. Each example contains comments that are delimited using standard C/C++ notation (/* */). Comments are more sparse in the later examples. Additionally, the relevant parts of the ARM are presented, as well as the relevant mapping tables of the AIM.

ASCII CL-file sources for the examples are also included in the comments.

WARNING!!!! The examples are incomplete. I am working on unifying them with the example CDS/Cone AP238 file. They should only be used as a reference. I am working to correct this.

WARNING!!! The examples 2, 3, and 7 do not define UNITs as per AP238. I am working on these examples.

Example 1: A Three – Axis Toolpath

/* A toolpath can be represented as a list of three axis coordinates, each of which contains the X, Y, and Z axis coordinates of the centerline of the cutter. Since this is the way that toolpaths are described in CL files, the translation to this format is straightforward
*/

/* Source CL-File for example */

```
/*
25 GOTO  /   - 6.0000, - 6.0000, -3.1900
26 GOTO  /   - 5.0000, + 6.0000, -3.1900
27 GOTO  /   + 6.0000, + 6.0000, -4.1900
*/

#4230=MACHINING_TOOLPATH('WS 2 TP 1','cutter location trajectory','','');
↑
#2504=ACTION_PROPERTY('basic curve','',#4230);
↑
#516=ACTION_PROPERTY_REPRESENTATION('basic curve','',#2504,#1509);
↓
#1509=REPRESENTATION($,(#3723),$);
↑
#3723=POLYLINE('WS 2 TP 1',(#5246,#5247,#5248));
    ↓
     #5246=CARTESIAN_POINT($,(-6.,-6.,-3.19));
  ↓
  #5247=CARTESIAN_POINT($,(-5.,6.,-3.19));
↓
#5248=CARTESIAN_POINT($,(6.,6.,-4.19));

/* ARM */
/*
toolpath
trajectory
cutter_location_trajectory
```

basiccurve
bounded_curve
*/

#4230
machining_toolpath <=
action_method
{ action_method.description 'cutter location trajectory' }
/* The machining toolpath is defined as a cutter location trajectory, which means that the toolpath that is defined represents the path of the centerline of the cutter.
*/

#2504
action_property.definition
{ action_property.name = 'basic curve' }
/* The path of the centerline of the cutter is a bounded_curve. The attribute 'definition' points to the machining_toolpath
*/

#516
action_property <-
action_property_representation.property
action_property_representation
action_property_representation.representation ->
representation
/* the action_property_representation links the action_property to its representation. The attribute 'property' points to the action_property. The attribute 'representation' points to the representation.
*/

#1509
representation.items[i]->
representation_item =>
geometric_representation_item
/* Allows a more general representation */

#3723
representation_item =>

geometric_representation_item =>
curve =>
bounded_curve
/* A polyline is a bounded curve, which is
a subtype of curve, which is
a subtype of a geometric_representation_item which is
a subtype of a representation_item.

10303-42 defines a polyline as a subtype of a bounded_curve
10303-42 defines a polyline as a list of cartesian_point
*/

#5246, #5247, #5248
/* 10303-42 defines a cartesian_point as list of points. In this context, the list is a length of three. Each of the three points in the list describes the X, Y, and Z axis positions, of the centerline of the cutter.
*/

Example 2: A Five – Axis Toolpath

/* A five-axis toolpath is similar to a three-axis toolpath. Like a three-axis toolpath, the cutter centerline path is also represented by a polyline. However, in a five-axis toolpath, a second polyline, of the same length as the cutter centerline path, describes the tool orientation as a unit vector. Again, this is a straightforward translation from a CL file
*/

/* Source CL-File for example */

```
/*
   37 GOTO  /   + 0.0000, + 0.4805, - 5.0728, +0.000000,  +0.258800,  +0.965900
   38 GOTO  /   - 0.0136, + 0.4803, - 5.0728, +0.007300,  +0.258700,  +0.965900
   39 GOTO  /   + 0.0200, + 0.4801, - 5.0728, +0.010800,  +0.258600,  +0.965900
*
```

/* Tool Tip Position */

```
#4470=MACHINING_TOOLPATH('WS 3 TP 51','cutter location trajectory','','');
  ↑
  #3706=ACTION_PROPERTY('basic curve','',#4470);
  ↑
  #1484=ACTION_PROPERTY_REPRESENTATION('basic curve','',#3706,#2477);
  ↓
  #2477=REPRESENTATION($,(#3969),$);
  ↓
  #3969=POLYLINE('WS 3 TP 51',(#9305,#9307,#9309));
      ↓
      #9305=CARTESIAN_POINT($,(0.,0.4805,-5.0728));
    ↓
    #9307=CARTESIAN_POINT($,(0.0136,0.4803,-5.0728));
  ↓
  #9309=CARTESIAN_POINT($,(0.02,0.4801,-5.0728));
```

/* Tool Vectors (Orientation) */
↑
#3707=ACTION_PROPERTY('tool axis','',#4470);
↑
#1485=ACTION_PROPERTY_REPRESENTATION($,$,#3707,#2478);
↓
#2478=REPRESENTATION($,(#3970),$);
↓
#3970=POLYLINE('WS 3 TP 51',(#9306,#9308,#9310,#9312,#9314,#9316));
↓
#9306=CARTESIAN_POINT($,(0.,0.2588,0.9659));
↓
#9308=CARTESIAN_POINT($,(0.0073,0.2587,0.9659));
↓
#9310=CARTESIAN_POINT($,(0.0108,0.2586,0.9659));

/* ARM */
/*
toolpath
trajectory
cutter_location_trajectory
basiccurve
bounded_curve

toolpath
trajectory
cutter_location_trajectory
its_toolaxis
basiccurve
bounded_curve

*/

#4470

machining_toolpath <=
action_method
{ action_method.description 'cutter location trajectory' }
/* The machining toolpath is defined as a cutter location trajectory, which means that the toolpath that is defined represents the path of the centerline of the cutter.
*/

#3706
action_property.definition
{ action_property.name = 'basic curve' }
/* The path of the centerline of the cutter is a bounded_curve. The attribute 'definition' points to the machining_toolpath
*/

#1484
action_property <-
action_property_representation.property
action_property_representation
action_property_representation.representation ->
representation
/* the action_property_representation links the action_property to its representation. The attribute 'property' points to the action_property. The attribute 'representation' points to the representation.
*/

#2477
representation.items[i]->
representation_item =>
geometric_representation_item
/* Allows a more general representation */

#3969
representation_item =>
geometric_representation_item =>
curve =>
bounded_curve
/* A polyline is a bounded curve, which is
a subtype of curve, which is
a subtype of a geometric_representation_item which is

```
      a subtype of a representation_item.

      10303-42 defines a polyline as a subtype of a bounded_curve
      10303-42 defines a polyline as a list of Cartesian_point
*/

#9305, #9307, #9309
/* 10303-42 defines a Cartesian_point as list of points.  In this context, the list is a length of
three.  Each of the three points in the list describe the X, Y, and Z axis positions, of the
centerline of the cutter.
*/

#3707
action_property.definition
{ action_property.name = 'tool axis' }
/* The orientation of the cutter is defined as a bounded_curve of unit vectors.  The attribute
'definition' points to the machining_toolpath
*/

#1485
action_property <-
action_property_representation.property
action_property_representation
action_property_representation.representation ->
representation
/* the action_property_representation links the action_property to its representation.  The
attribute 'property' points to the action_property.  The attribute 'representation' points to the
representation.
*/

#2478
representation.items[i]->
representation_item =>
geometric_representation_item
/* Allows a more general representation */

#3970
representation_item =>
geometric_representation_item =>
```

curve =>
bounded_curve
/* A polyline is a bounded curve, which is
a subtype of curve, which is
a subtype of a geometric_representation_item which is
a subtype of a representation_item.

10303-42 defines a polyline as a subtype of a bounded_curve
10303-42 defines a polyline as a list of Cartesian_point

In this case, the bounded curve represents the tool orientation as a series of unit vectors
*/
#9306, #9308, #9310
/* 10303-42 defines a Cartesian_point as list of points. In this context, the list is a length of three. Each of the three points in the list describe the i, j, and k tool orientation cosines of the cutter.
*/

Example 3: Feedrate for a toolpath

/* In AP238, there are many ways to define the feedrate for a tool path. Feedrates can be attached to a workingstep, toolpath, or expressed as a ratio of a base feedrate. Feedrates can be expressed as constants, polylines, or b-splines. In this example, the feedrate is expressed directly as a constant. It is attached to a toolpath.
*/

/* Source CL-File for example */

```
/*
22 FEDRAT/   300.0000
*/
```

/* ARM */
/*
toolpath
its_technology
technology
feedrate
*/

```
#4230=MACHINING_TOOLPATH('WS 2 TP 1','cutter location trajectory','','');
↑
#1510=MACHINING_TECHNOLOGY_RELATIONSHIP('technology',$,#4230,#4983);
↓
#4983=MACHINING_TECHNOLOGY($,'milling',$,$);
↑
#1=ACTION_PROPERTY('feedrate','',#4983);
↑
#518=ACTION_PROPERTY_REPRESENTATION($,$,#1,#1512);
↓
#1512=MACHINING_FEED_SPEED_REPRESENTATION('feed speed',(#20),$);
```

↓
#20=MEASURE_REPRESENTATION_ITEM('feed speed',MEASURE_WITH_UNIT(50.),$);

#4230
/* A machining_toolpath is a subtype of an action_method *’

#1510
machining_toolpath <=
action_method <-
action_method_relationship.relating_method
action_method_relationship
{ action_method_relationship =>
machining_technology_relationship }
action_method_relationship.related_method ->
action_method =>
machining_technology
/* The relationship between a toolpath and its technology is established, with the relating_method attribute pointing to the toolpath and the related_method attribute pointing to the technology */

#1
machining_technology <=
action_method
characterized_action_definition = action_method
characterized_action_definition <-
action_property.definition
{ action_property.name = ‘feedrate’ }
/* We’re defining machining technology. The attribute characterized_action_definition points to the technology */

#518
action_property <-
action_property_representation.property
action_property_representation

```
action_property_representation.representation ->
{ representiation =>
machining_feed_speed_representation
representation
/* The relationship between the action_property and the representation is established */

#1512
{ representation.name ' 'feed speed' }
representation.items[i] ->
{ represention_item.name ' 'feed speed' }
representation_item =>
measure_representation_item <=
measure_with_unit
{ measure_with_unit_value_component ->
measure_value

#20
measure_value = numeric_measure
numeric_measure }
```

Example 4: Priority for a toolpath

```
#4230=MACHINING_TOOLPATH('WS 2 TP 1','cutter location trajectory','','');
↑
#2508=ACTION_PROPERTY('priority','',#4230);
↑
#519=ACTION_PROPERTY_REPRESENTATION($,$,#2508,#1512);
↓
#1512=REPRESENTATION($,(#20),$);
↓
#20=DESCRIPTIVE_REPRESENTATION_ITEM('WS 2 TP 2','required');

/* Source CL-File for example */
/*
There is no equivalent toolpath priority in a CL file.  All toolpaths in either a CL file or AP238
CC1 file are mandatory
*/

/* ARM */
toolpath
its_priority
boolean

/* AIM */
#4230
machining_toolpath <=
action_method
characterized_action_definition = action_method
/* An action method is being defined for the toolpath.
*/

#2508
characterized_action_definition <-
action_property.definition
```

```
{ action_property.name = 'priority' }
/* A toolpath priority is being established.  Toolpath priorities are mostly irrelevant in CC1,
since a description of the part features is not being transmitted.  Toolpath priorities allow the
programmer to chose whether a capable CNC is allowed to generate its own toolpaths or force it to
accept the programmed toolpaths

The attribute 'definition' points to the relevant toolpath
*/

#519
action_property <-
action_property_representation.property
action_property_representation
action_property_representation.representation->
representation
/* The action_property is linked by the 'property' attribute.
The representation is linked by the 'representation' attribute.
*/

#1512
representation.items[i]->
/* The attribute 'items' points to the representation_item.
*/

#20
representation_item =>
descriptive_representation_item
descriptive_representation_item.description
{ (descriptive_representation_item.description = 'required' )
(descriptive_representation_item.description = 'suggested' )}
/* A descriptive_representation_item is a subtype of a representation_item
When the description is 'required', the explicitly defined toolpath must be followed.
When the description is 'suggested', the explicitly defined toolpath may be ignored.
*/
```

Example 5: Movement type for a toolpath

```
#4230=MACHINING_TOOLPATH('WS 2 TP 1','cutter location trajectory','','');
↑
#2507=ACTION_PROPERTY('movement type','',#4230);
↑
#518=ACTION_PROPERTY_REPRESENTATION($,$,#2507,#1511);
↓
#1511=REPRESENTATION($,(#19),$);
↓
#19=DESCRIPTIVE_REPRESENTATION_ITEM('WS 2 TP 2','trajectory path');


/* Source CL-File for example */
/*
There is no equivalent movement type in a CL file.  All toolpaths are represented in CC1 as a
trajectory path
*/


/* ARM */
toolpath
its_type
toolpath_type

#4230
machining_toolpath <=
action_method
characterized_action_definition = action_method

#2507
characterized_action_definition <-
action_property.definition
{ action_property.name = 'movement type' }
```

```
#518
action_property <-
action_property_representation.property
action_property_representation
action_property_representation.representation ->
representation

#1511

representation.items[i] ->
representation_item ->
descriptive_representation_item

#19
descriptive_representation_item.description
{ ( descriptive_representation_item.description = 'approach' )
( descriptive_representation_item.description = 'lift' )
( descriptive_representation_item.description = 'connect' )
( descriptive_representation_item.description = 'non-contact' )
( descriptive_representation_item.description = 'contact' )
( descriptive_representation_item.description = 'trajectory path' ) }
```

Example 6: Relationship of toolpaths to workingstep

```
#4973=FREEFORM_MILLING_OPERATION('Workingstep 2',$,$,$);
         ↑
          #4727= MACHINING_TOOLPATH_SEQUENCE_RELATIONSHIP ('',$,#4973,#4229,0.);
          ↓
          #4229=MACHINING_TOOLPATH('WS 2 TP 0','cutter location trajectory','','');
        ↑
        #4728= MACHINING_TOOLPATH_SEQUENCE_RELATIONSHIP ('',$,#4973,#4230,1.);
        ↓
```

```
   #4230=MACHINING_TOOLPATH('WS 2 TP 1','cutter location trajectory','','');
↑
 #4729= MACHINING_TOOLPATH_SEQUENCE_RELATIONSHIP ('',$,#4973,#4231,2.);
↓
 #4231=MACHINING_TOOLPATH('WS 2 TP 2','cutter location trajectory','','');
```

```
/* Source CL-File for example */
/*
There is no direct equivalent in a CL file.  However, a new workingstep must be generated in an
AP238 CC1 file whenever a change of cutter is required.  It might be a good idea to also create a
new workingstep whenever a new operation is called in the source CL file.
*/
```

#4973

#4229, #4230, #4231

```
/* The toolpaths */
```

```
/* ARM */
operation
its_toolpath
toolpath_list
its_list
toolpath
```

Example 7: Relationship of spindle speed to workingstep

```
#4973=FREEFORM_MILLING_OPERATION('Workingstep 2',$,$,$);
↑
#4981=MACHINING_TECHNOLOGY_RELATIONSHIP ('',$,#4973,#4475);
↓
#4475=MACHINING_TECHNOLOGY($,'milling',$,$);
↑
#2498=ACTION_PROPERTY('spindle',$,#4475);
↑
#511=ACTION_PROPERTY_REPRESENTATION($,$,#2498,#1504);
↓
#1504=MACHINING_SPINDLE_SPEED_REPRESENTATION('spindle speed',(#3979),$);
↓
#3979=MEASURE_REPRESENTATION_ITEM('rotational speed',NUMERIC_MEASURE(3600.),
$);
```

```
/* Source CL-File for example */
/*
The connection between a spindle speed command in a CL file and an AP238 file is indirect.  In a
CL file, spindle speeds and direction of rotation are treated as modal functions that occur
sequentially as the file is executed.  In an AP238 CC1 file, the spindle speed and direction of
rotation are attached to a workingstep or tool path.

 */
/*
6 SPINDL/ ON, 3600.0
*/
```

#4981

```
#4475
machining_technology <=
action_method
characterized_action_definition=action_method

#2498
characterized_action_definition <-
action_property.definition
{ action_property.name = 'spindle' }

#511
action_property <-
action_property_representation.property
action_property_representation
action_property_representation.representation ->
{ representation =>
machining_spindle_speed_representation }

#1504
machinining_spindle_speed_representation
{ representation.name = 'spindle speed' }
representation_items[i] ->

#3979
{ representation_item.name = 'rotational speed' }
representation_item =>
measure_representation_item <=
measure_with_unit
{ measure_with_unit.value_component ->
measure_value
measure_value = numeric_measure
numeric_measure }
measure
```

Example 8: Relationship of coolant to workingstep

```
#4973=FREEFORM_MILLING_OPERATION('Workingstep 2',$,$,$);
↑
#4980=MACHINING_FUNCTIONS_RELATIONSHIP('machine functions',$,#4973,#4722);
↓
#4722=MACHINING_FUNCTIONS($,'milling',$,$);
↑
#2497=ACTION_PROPERTY('coolant',$,#4722);
↑
#510=ACTION_PROPERTY_REPRESENTATION($,$,#2497,#1503);
↓
#1503=REPRESENTATION('coolant',(#16),$);
↓
#16=DESCRIPTIVE_REPRESENTATION_ITEM($,'coolant on');
```

```
/* Source CL-File for example */
/*
The connection between a coolant command in a CL file and an AP238 CC1 file is indirect.  In a CL
file, coolant commands are treated as modal functions that occur sequentially as the file is
executed.  In an AP238 CC1 file, the coolant commands are attached to a workingstep or tool path.

 */
/*
7 COOLNT/ ON
*/



/* ARM */

executable
workingstep
machining_workingstep
```

```
its_operation
machining_operation
its_machine_functions
milling_machine_functions
coolant
```

```
#4973
freeform_milling_operation <=
milling_type_operation <=
machining_operation

#4980
machining_operation <=
action_method <-
action_method_relationship.relating_method
action_method_relationship
{ action_method_relationship =>
machining_functions_relationship }
action_method_relationship.related_method ->
action_method =>
machining_functions

#4722
machining_functions <=
action_method
{ action_method.description = 'milling' }

machining_functions <=
action_method
characterized_action_definition = action_method

#2497
```

```
characterized_action_definition <-
action_property.definition
{ action_property.name = 'coolant' }

#510
action_property <-
action_property_representation.property
action_property_representation
action_property_representation.representation ->
representation

#1503
representation.items[i] ->
representation_item =>
descriptive_representation_item

#16
descriptive_representation_item.description
{ (descriptive_representation_item.description.description = 'coolant on' )
(descriptive_representation_item.description = 'coolant off' ) }
```

Example 9: Relationship of tool to workingstep

```
#4973=FREEFORM_MILLING_OPERATION('Workingstep 2',$,$,$);
↑
#12=MACHINING_TOOL('3',$,(#4973),#10);
↓
#10=ACTION_RESOURCE_TYPE('cutting tool');

/* Source CL-File for example */
/*
The connection between a tool change in a CL file and an AP238 CC1 file is indirect.  In a CL
file, tool changes are treated as modal functions that occur sequentially as the file is executed.
In an AP238 CC1 file, the tool is attached to a workingstep.

 */
/*
    5 LOADTL/ 3
*/
```

Example 10: Relationship of workplan to workingstep

```
#5233=MACHINING_WORKPLAN('WRKPLAN-01',$,$,$);
↑
#4726= MACHINING_PROCESS_SEQUENCE_RELATIONSHIP ('Workingstep 2',$,#5233,#5230,0.);
↓
#5230=MACHINING_WORKINGSTEP($,$,$,$);
↑
#4979=MACHINING_OPERATION_RELATIONSHIP ('',$,#5230,#4973);
↓
#4973=FREEFORM_MILLING_OPERATION('Workingstep 2',$,$,$);


/* Source CL-File for example */
/*
There is no direct equivalent in a CL file.  However, a new workingstep must be generated in an
AP238 CC1 file whenever a change of cutter is required.  It might be a good idea to also create a
new workingstep whenever a new operation is called in the source CL file.
*/


/* ARM */
```

Example 11: Root relationship

```
#5240=APPLICATION_CONTEXT(
'Application protocol for the exchange of CNC data');
↑
#5239=PRODUCT_CONTEXT($,#5240,$);
↑
#5237=MACHINING_PROJECT($,$,$,(#5239));
↑
#5236=PRODUCT_DEFINITION_FORMATION($,$,#5237);
↑
#5235=PRODUCT_DEFINITION($,$,#5236,$);
↑
#5234=PROCESS_PRODUCT_ASSOCIATION($,$,#5235,#5232);
↓
#5232=PRODUCT_DEFINITION_PROCESS('machining','',#5233,'');
↓
#5233=MACHINING_WORKPLAN('WRKPLAN-01',$,$,$);
```

```
/* Source CL-File for example */
/*
There is no direct equivalent in a CL file.  A CL file is a machining workplan.  It does not
contain any other data,
*/



/* ARM */

project
project_to_workplan (as main_workplan)
```

#5235
product_definition_formation <-

```
product_definition.formation
product_definition
characterized_product_definition = product_definition

#5234
characterized_product_definition <-
process_product_association.defined_product
process_product_association
process_product_association.process ->
process_definition_process =>
action

#5232
{ action.name = 'machining' }
action.chosen_method ->
action_method =>
machining_process_executable =>
machining_workplan
```

ENTITY REFERENCE

Purpose

This chapter describes the syntax and required semantics of each entity in the AP-238 CC1 OMAC subset.

EXPRESS

In STEP each entity has a formal definition described by the EXPRESS language. An EXPRESS entity is similar to a record in a database and a STRUCT in a programming language. We will give the EXPRESS definition for each entity and then a description of each attribute in the entity.

Special Conventions

The full EXPRESS definition of each entity is given in the ISO 10303-238 standard. To make reading and understanding the definitions easier in this reference we will compromise the inheritance model in this document. Specifically, if an entity has inherited attributes they will be put directly into the entity and shown in **bold case** in this document.

The following example illustrates. Here is the official EXPRESS definition for a CARTESIAN_POINT entity as it appears in AP-238.

```
ENTITY cartesian_point
      SUBTYPE OF (point);
      coordinates : LIST [1:3] OF length_measure;
END_ENTITY;
```

As per this definition, cartesian_point is a subtype of point. If you look at the EXPRESS definitions in AP-238 you will find that point is a subtype of geometric_representation_item and it is a subtype of representation_item. From all these sources the cartesian_point entity inherits an additional attribute called name. Rather than make you hunt through the inheritance model searching for these inherited attributes we are going to list them directly in the entity and show that we have made the change using bold case. Therefore, in this document the cartesian_point entity will be listed as follows:

```
ENTITY cartesian_point
      SUBTYPE OF (point);
      name : label;
      coordinates : LIST [1:3] OF length_measure;
END_ENTITY;
```

Also because you may not know that the EXPRESS defines a LABEL is a STRING and a LENGTH_MEASURE is a REAL we will change the attribute definitions to the underlying type as follows:

```
ENTITY cartesian_point
      SUBTYPE OF (point);
      name : STRING;
      coordinates : LIST [1:3] OF REAL;
END_ENTITY;
```

In summary be aware when you see a definition in bold, it has been simplified from the AP-238 document for this presentation.

Meaning of terms

If the text says that something is required or that the "AP-238 standard requires" then the property or value being described is something required by the AP-238 standard and must not be violated under any circumstances.

If the text says that something is recommended or that the "AP-238 CC1 OMAC subset recommends" then the property or value being described is an addition that is required by the AP-238 CC1 OMAC subset in order to make implementation of the standard easier.

If the text says that something is suggested or that the "AP-238 CC1 OMAC subset suggests" then the property or value being described is a suggestion that should be used as a default when no other value is available.

ENTITY action_property

Action property entities are used to represent process parameters. The value of the property is described by a representation that is connected to the property by an action_property_representation entity.

Action property entities describe properties for the entity whose address is stored in their definition attribute.

Most, but not all, of the ARM attributes in the ISO 14649 model are stored as action_properties in the AP-238 AIM.

```
ENTITY action_property;
    name        : STRING;
    description : STRING;
    definition  : characterized_action_definition;
END_ENTITY; -- 10303-49: process_property_schema
```

There are five types of action properties allowed in the CC1 subset. The type of the property is defined by the name attribute. There are five values allowed for this attribute. In the AP-238 CC1 subset the values can be divided into two groups. The first group defines properties whose value is set for every instance.

'basic curve'	if the property defines the path geometry of a machining_toolpath entity
'tool axis'	if the property defines the axis geometry of a machining_toolpath entity
'feedrate'	if the property defines the federate of a machining_technology entity
'spindle'	if the property defines the federate of a machining_technology entity
'coolant'	if the property defines the coolant of a machining_functions entity

The second group defines properties that are the same for every instance.

'movement type'	if the property defines the movement type of a machining_toolpath entity
'priority'	if the property defines the priority of a machining_toolpath entity.

In the CC1 subset the value of the movement type property is always “trajectory path”. Other values allowed in AP-238 but not the CC1 subset are: “approach”, “lift”, “connect”, “non-contact”, “contact”. These values are hints to the controller about the type of path being described. For more information see ISO 14649-10.

In the CC1 subset the value of the priority property is always “required”. The other value allowed in AP-238 but not the CC1 subset is: “suggested”. If the priority is required then the tool path must be used by the control. If the priority is suggested then the controller may generate its own tool path from the feature description associated with the working step.

In the AP-238 CC1 OMAC subset the description attribute should be set to the same value as the name of the machining workingstep that contains this property or the name of the machining toolpath that contains this property.

The entity addressed by the definition attribute depends on the type of the property. If the action_property defines a federate or spindle speed then the definition attribute will address a machining_technology entity. If the action_property defines a coolant value then the definition attribute will address a machining_functions entity. In all other cases the definition will address a machining_toolpath entity.

ENTITY action_property_representation

Each action property is connected to its representation by an action property representation entity.

```
ENTITY action_property_representation;
        name           : STRING;
        description    : STRING;
        property       : action_property;
        representation : representation;
END_ENTITY;
```

The name of the action_property_representation is not defined by the standard. For the AP-238 CC1 subset it should be set to the same value as the name of the action_property.

The description attribute is not defined by the standard. For the AP-238 CC1 OMAC subset it should have the same value as the description of the action_property.

The property attribute should be set to the address of the property being described.

The representation should be set to the address of the representation of the property. If the value of the representation is constant then the representation can be shared by many properties. See the description of the representation entity for more details.

ENTITY action_resource_type

An action_resource_type entity qualifies a machining_tool entity with the type of the cutting tool.

```
ENTITY action_resource_type;
   name : STRING;
END_ENTITY;
```

There are three kinds of cutting tools allowed in AP-238. The name attribute is used to select the type. The values allowed for this attribute are: 'milling cutting tool' and 'turning cutting tool'. Only 'milling cutting tool' is allowed in the AP-238 CC1 subset. The action_resource_type entity for each type can be shared by all the machining tool entities in a file.

ENTITY application_context

The application_context entity is a header entity. In the AP-238 CC1 OMAC subset it is used to show that this file conforms to the conventions described in the AP-238 OMAC subset. If a file has an application_context entity with the application name described here then it should conform to these conventions as well as the rules described in the AP-238 CC1 definition.

The conventions defined in the AP-238 CC1 OMAC subset have been defined to make it easier for organizations and vendors to get started with AP-238 CC1.

The application_context entity is addressed by a product_context entity and a product_definition_context entity. These two entities describe information about the CNC program in this file. There should only be one instance of all three entities in an AP-238 CC1 OMAC subset file.

```
ENTITY application_context;
    application : STRING;
END_ENTITY ;
```

The AP-238 CC1 OMAC subset recommends setting the application attribute to the value `'AP-238 CC1 OMAC Subset Version 1.0'`. A version number is included in case the OMAC group decides to define multiple versions of the subset.

ENTITY application_protocol_definition

The application_protocol definition entity is a header entity that describes information about the version of the ISO 10303-238 Application Protocol that defines this file.

The application_protocol_definition entity addresses the application_context entity that describes information about the file. There should only be one instance of each entity in an AP-238 CC1 OMAC subset file.

```
ENTITY application_protocol_definition;
    status                                  : STRING;
    application_interpreted_model_schema_name : STRING;
    application_protocol_year               : INTEGER;
    application                             : application_context;
END_ENTITY;
```

The AP-238 OMAC CC1 subset requires the status attribute to be set to the value `'Draft International Standard'`.

The AP-238 OMAC CC1 subset requires the application_interpreted_model_schema_name attribute to be set to the value `'integrated_cnc_schema'`.

The AP-238 OMAC CC1 subset requires the application_protocol_year attribute to be set to the integer value 2004.

The standard requires the application attribute to be set to the address of an application_context entity. In the AP-238 CC1 OMAC subset this entity will define the version number of the AP-238 CC1 subset that defines the conventions that are being applied to the file in addition to those described by AP-238.

ENTITY cartesian_point

Cartesian_points are used to describe coordinates for the x, y, z components of tool paths and orientations for the i, j, k components of tool axes.

If the cartesian_point describes the x, y, z coordinates for a tool path then it must describe the absolute coordinates of that point relative to the origin defined in the set-up. No transformations are to be defined or applied to these coordinates by other structures in the file. The units of the coordinates are defined in the representation that contains the entity. In the AP-238 CC1 subset the units are inches.

If the cartesian_point describes an orientation for a tool axis then that axis must be described as a unit vector.

A tool path is defined by a sequence of cartesian_points whose addresses are stored in a representation entity. These representation entities are connected to a machining_toolpath entity by action_property and action_property_representation entities. See the description of these entities.

If a machining_toolpath is 3-axis then it will use only one representation for its tool path and this representation will be connected to the machining tool path by an action_property with the value “basic curve” in its name attribute.

If a machining_toolpath is 5-axis then it will use two representations for its tool path. The first representation will be as described for a 3-axis toolpath. The second representation will be connected to the machining tool path by an action_property with the value “tool axis” in its name attribute. The number of cartesian_points in the two representations must be identical.

```
ENTITY cartesian_point
  SUBTYPE OF (point);
      name        : STRING;
      coordinates : LIST [1:3] OF REAL;
END_ENTITY;
```

The name attribute can be any value. We recommend setting it to a meaningful value if one has been supplied by the end-user and to the empty string otherwise (`''`).

The coordinates must contain three values that represent x, y and z or i, j and k depending on the usage of the point.

ENTITY descriptive_representation_item

Descriptive_representation_items are used to describe text values for simple properties such as Coolant (which can have the value “on” or “off”).

```
ENTITY descriptive_representation_item
      SUBTYPE OF (representation_item);
      name        : STRING;
      description : STRING;
END_ENTITY;
```

All of the descriptive representation items in the CC1 subset describe constant values that can be shared by multiple entities. Four entities need to be created. Each entity should have the value 'shared constant' in its name attribute.

The description is the value of the item depends on the property being described. The four description values allowed in the CC1 subset are:

'trajectory path'	to indicate that this tool path is defined by tool center point data.
'required'	to indicate that this tool path is required for this working step.
'coolant off'	to indicate that the coolant is to be turned off when a command containing this item is executed
'coolant on'	to indicate that the coolant is to be turned on when a command containing this item is executed

Each descriptive_representation_item must belong to a representation entity. For more information see the description of the representation entity.

ENTITY freeform_milling_operation

Freeform_milling_operations are used to describe operations for workingsteps that perform freeform_milling. In the AP-238 CC1 OMAC subset this is the only kind of operation allowed.

A freeform_milling_operation is defined by a sequence of tool paths. Each toolpath is defined by a machining_toolpath entity that will be connected to the freeform_milling_operation entity by a machining_toolpath_sequence_relationship entity. The position of each toolpath in the sequence is defined by a sequence_number in the machining_toolpath_sequence_relationship entity.

At minimum you must define a new freeform_milling_operation each time the cutting tool is changed and a new machining_toolpath entity each time the feedrate is changed. If possible, however, you should define a new freeform_milling_operation each time the feature changes as well because this will make the data more compatible with the other AP-238 conformance classes.

A freeform_milling_operation entity is connected to a description of its cutting tool by a machining_tool entity.

A freeform_milling_operation entity is connected to a description of its machine functions by a machining_functions_relationship entity. The machine_functions entity will describe the coolant (on or off) value of the operation.

A freeform_milling_operation entity is connected to its workingstep by a machining_operation_relationship entity. In the OMAC AP-238 CC1 subset there will be one workingstep for each freeform_milling_operation and vice versa.

```
ENTITY freeform_milling_operation
  SUBTYPE OF (milling_type_operation);
      name        : STRING;
      description : STRING;
      consequence : STRING;
      purpose     : STRING;
END_ENTITY;
```

We recommend you use the name attribute to describe the working step number that is using this function. For example, `'WS 2'` means this entity describes the freeform_milling_operation for Workingstep 2.

We recommend that the description attribute to a value that is meaningful to the end user or the empty string (").

The consequence attribute is not used in our CC1 subset. We recommend setting its value to the empty string (").

The purpose attribute is not used in our CC1 subset. We recommend setting its value to the empty string ('').

ENTITY machining_feed_speed_representation

A machining_feed_speed_representation is a type of representation that is restricted to holding feed speed values.

```
ENTITY machining_feed_speed_representation
      SUBTYPE_OF (representation);
    name             : STRING;
    items            : SET [1:?] OF representation_item;
    context_of_items : representation_context;
END_ENTITY;
```

The standard gives a list of required values for the name attribute. In the AP-238 CC1 OMAC subset this range is further restricted to be the value `'feed speed'`.

The AP-238 CC1 OMAC subset recommends the items attribute contains the address of a single measure_representation_item that describes the feed speed value.

The AP-238 CC1 OMAC subset recommends the context_of_items attribute addresses the following entity because the feed speed units are described in the measure_represenation_item.
#9904=REPRESENTATION_CONTEXT('representation','units NOT necessary');

ENTITY machining_functions

In AP-238 machining_functions entities are used to describe the machine functions that must be set for an operation or a toolpath.

In the AP-238 CC1 OMAC subset, machining functions are set for freeform_milling operations only and the only function set is the coolant value to coolant_on and coolant_off.

A machining_functions entity is connected to a freeform_milling_operation entity by a machining_functions_relationship entity.

A machining_functions entity is connected to a representation of the coolant value by an action_property entity.

```
ENTITY machining_functions
  SUBTYPE OF (action_method);
      name        : STRING;
      description : STRING;
      consequence : STRING;
      purpose     : STRING;
END_ENTITY;
```

We recommend you use the name attribute to describe the working step number that is using this function. For example, 'WS 2' means this entity describes machining_functions for Workingstep 2.

In the CC1 subset the description must have the following value:

```
'milling'
```

The consequence attribute is not used in our CC1 subset. We recommend setting its value to the empty string (").

The purpose attribute is not used in our CC1 subset. We recommend setting its value to the empty string (").

ENTITY machining_functions_relationship

In the AP-238 CC1 subset a machining_functions_relationship entity is used to connect a freeform_milling_operation to a machining_functions entity so that the latter can describe a coolant value for the former.

```
ENTITY machining_functions_relationship
  SUBTYPE OF (action_method_relationship);
      name            : STRING;
      description     : OPTIONAL STRING;
      relating_method : action_method;
      related_method  : action_method;
END_ENTITY;
```

The standard requires the name attribute to contain the value `'machine functions'.`

We recommend setting the value of the description attribute to `'coolant value'`.

The OMAC AP-238 CC1 subset requires the relating_method attribute to contain the address of a freeform_milling_operation entity.

The OMAC AP-238 CC1 subset requires the related_method attribute to contain the address of a machining_functions entity.

ENTITY machining_operation_relationship

A machining_operation_relationship entity is used to connect a machining_operation to a machining_workingstep entity so that the former can describe an operation for the latter. In the AP-238 subset the machining_operation entity must be a freeform_milling_operation entity.

```
ENTITY machining_operation_relationship
  SUBTYPE OF (action_method_relationship);
      name             : STRING;
      description      : OPTIONAL STRING;
      relating_method  : action_method;
      related_method   : action_method;
END_ENTITY;
```

We recommend setting the name attribute to the same value as the name attribute of the machining_workingstep.

We recommend setting the description attribute to end user data or the empty string.

The relating_method attribute must contain the address of the machining_workingstep entity.

The related_method attribute must contain the address of the freeform_milling_operation entity.

ENTITY machining_process_sequence_relationship

A machining_process_sequence_relaitionship entity is used to define a process sequence for a machining workplan.

In the AP-238 CC1 OMAC subset a machining_process_sequence_relationship entity is always used to connect a machining_workingstep entity to a machining_workplan entity. In the full AP-238 standard there a machining_workplan can contain nested machining_workplans as well as machining_workinsteps. In the AP-238 CC1 OMAC subset there will only be one machining_workplan entity and one machining_process_sequence entity in the file.

```
ENTITY machining_process_sequence_relationship
  SUBTYPE OF (sequential_method);
      name              : STRING;
      description       : OPTIONAL STRING;
      relating_method   : action_method;
      related_method    : action_method;
      sequence_position : REAL;
END_ENTITY;
```

We recommend setting the value of the name attribute to same value as the machining_workingstep name.

We recommend setting the description attribute to end user data or the empty string.

The standard requires the relating_method attribute to contain the address of the machining_workplan that contains the machining_workingstep entity.

In the AP-238 CC1 OMAC subset, the related_method attribute will contain the address of the machining_workingstep entity.

The standard requires the sequence_position attribute to contain the position of this machining_workingstep entity in the sequence of machining_workingstep entities belonging to the machining_workplan entity. This value must be unique within the set of machining_process_sequence_relationship entities that address a particular machining_workplan entity. The value is defined as a real to allow for the insertion of new workingsteps into an existing list.

ENTITY machining_project

The machining_project entity is the root entity of an AP-238 file. A machining project entity is described in three parts.

The machining_project entity will be addressed by a product_definition_formation entity that is addressed by a product_definition entity. The product_definition_formation entity will describe the version number of this project. The product definition entity will lead to the workplan that contains the workingsteps that define the machining process.

In the OMAC AP-238 CC1 subset there will only be one product_definition_formation entity for each machining_project entity and there will only be one product_definition entity for each product_definition_formation entity.

```
ENTITY machining_project
          SUBTYPE OF (product);
    id                  : STRING;
    name                : STRING;
    description         : OPTIONAL STRING;
    frame_of_reference  : SET [1:?] OF product_context;
END_ENTITY;
```

The OMAC AP-238 CC1 subset recommends setting the id attribute to the organizations identifier for this project.

The OMAC AP-238 CC1 subset recommends setting the name attribute to the organizations informal name for the project if there is one and the same value as the id attribute otherwise. In the OMAC AP-238 CC1 subset this attribute will be duplicated in several other entities.

The OMAC AP-238 CC1 subset recommends setting the description attribute to end user data or the empty string.

The standard requires the frame_of_reference attribute to be set to the address of a product_context entity that indicates that this is an AP-238 CC1 OMAC subset file.

ENTITY machining_project_workpiece_relationship

A machining_porject_workpiece_relationship entity is used to connect a machining project to the product describing the workpiece that is going to be made by the project.

The standard requires the machining_project_workpiece_relationship entity to address the product_definition entity for the machining_project in its relating attribute and the product_definition entity for the workpiece in its related attribute.

In the AP-238 CC1 OMAC there will only be one machining_project product and one workpiece product. In AP-238 there can be many other products representing the stock, the fixtures and the tools.

```
ENTITY machining_project_workpiece_relationship
  SUBTYPE OF (product_definition_relationship);
      id                              : STRING;
      name                            : STRING;
      description                     : OPTIONAL STRING;
      relating_product_definition     : action_method;
      related_product_definition      : action_method;
END_ENTITY;
```

We recommend setting the value of the id attribute to same value as the machining_project name.

We recommend setting the value of the name attribute to same value as the workpiece product name.

We recommend setting the description attribute to end user data or the empty string.

The standard requires the relating_method attribute to contain the address of the product_definition entity for the machining_project.

The standard requires the related_method attribute to contain the address of the product_definition entity for the workpiece.

ENTITY machining_spindle_speed_representation

A machining_spindle_speed_representation is a type of representation that is restricted to holding spindle speed values.

```
ENTITY machining_spindle_speed_representation;
    name             : STRING;
    items            : SET [1:?] OF representation_item;
    context_of_items : representation_context;
END_ENTITY;
```

The standard gives a list of required values for the name attribute. In the AP-238 CC1 OMAC subset this range is further restricted to be the value `'spindle speed'`.

The AP-238 CC1 OMAC subset recommends the items attribute contains the address of a single measure_representation_item that describes the speed value.

The AP-238 CC1 OMAC subset recommends the context_of_items attribute addresses the following entity because the spindle speed units are described in the measure_representation_item.
#9904=REPRESENTATION_CONTEXT('representation','units NOT necessary');

ENTITY machining_technology

Machining_technology entities are used to describe the feed rate and spindle speed of a tool path.

A machining_technology entity is connected to a machining_toolpath entity by a machining_technology_relationship entity.

A machining_technology entity is connected to a representation of the feedrate by an action_property entity.

A machining_technology entity is connected to a representation of the spindle speed by an action_property entity.

```
ENTITY machining_technology
  SUBTYPE OF (action_method);
      name        : STRING;
      description : STRING;
      consequence : STRING;
      purpose     : STRING;
END_ENTITY;
```

We recommend you use the name attribute to describe the working step number and the tool path number that is using this feed and speed. For example, 'WS 2 TP 1' means this entity describes machining_technology attributes for Workingstep 2, tool path 1.

In the CC1 subset the description must have the following value:

`'milling'`

The consequence attribute is not used in our CC1 subset. We recommend setting its value to the empty string.

The purpose attribute is not used in our CC1 subset. We recommend setting its value to the empty string.

ENTITY machining_technology_relationship

In the AP-238 CC1 subset a machining_technology_relationship entity is used to connect a machining_toolpath entity to a machining_technology entity so that the latter can describe a feedrate for the former.

```
ENTITY machining_technology_relationship
  SUBTYPE OF (action_method_relationship);
      name            : STRING;
      description     : OPTIONAL STRING;
      relating_method : action_method;
      related_method  : action_method;
END_ENTITY;
```

We recommend setting the name value to the value of the name attribute of the machining_toolpath entity.

We recommend setting the value of the description attribute to `'feed and speed values'`.

The relating_method attribute must contain the address of the machining_toolpath entity.

The related_method attribute must contain the address of the machining_technology entity.

ENTITY machining_tool

Machining_tool entities are used to represent cutting tools. There should be one of these entities for each cutting tool in the CNC program.

A machining_tool is attached to the operations that use that tool via its usage attribute. There can be multiple operations that use the same tool.

```
ENTITY machining_tool
      SUBTYPE OF (action_resource);
      name        : STRING;
      description : OPTIONAL STRING;
      usage       : SET [1:?] OF supported_item;
      kind        : action_resource_type;
END_ENTITY;
```

Use the name attribute to give the tool a meaningful name. If the AP-238 CC1 file is being created from an APT CL file this should be the tool number e.g. ‘1’. If the AP-238 file is being created from a CAM system then it should be the enterprises formal name for the tool.

The description attribute is optional. If used this attribute should be set to an informal tool name that will be meaningful to the operator.

The usage attribute addresses the operations that use this tool. In the AP-238 CC1 subset these will be freeform_milling operations. In an AP-238 CNC program, a workingstep is defined by a feature and an operation. Only the operation is described in the CC1 subset and its most important data will be a toolpath that is connected to the operation by other entities.

The kind attribute describes the kind of the cutting tool. For more information see the action_resource_type entity.

ENTITY machining_toolpath

Machining_toolpath entities are used to describe tool paths.

A machining_tool path entity is connected to its operation by a machining_toolpath_sequence_relationship entity. In the AP-238 CC1 OMAC subset this operation will be a freeform milling operation. The position of this toolpath entity in the list of toolpaths belonging to the freeform_milling entity will be shown by the value of the sequence_position attribute of the machining_toolpath_sequence_relationship entity.

A machining_toolpath entity is connected to each of its properties by an action_property entity.

The most important property is the geometry of the toolpath. This property will be represented by an action_property entity with the value “basic curve” it its name attribute.

If the toolpath is 5-axis then the machining_toolpath entity will have an equally important property describing the geometry of the axes. This property will be represented by an action_property entity with the value “tool axis” it its name attribute.

A machining_toolpath entity is also required to have two properties describing the priority of the toolpath and the movement type of the toolpath. In the AP-238 CC1 subset these two properties always have the same value. For more details see the description of the action_property entity.

A machining_tool_path entity will be connected to a machining_technology entity describing the spindle speed and feedrate of the toolpath by a machining_technology_relationship entity.

```
ENTITY machining_toolpath
      SUBTYPE OF (action_method);
      name        : STRING;
      description : STRING;
      consequence : STRING;
      purpose     : STRING;
END_ENTITY;
```

We recommend using the name attribute to describe the working step number and the tool path number. For example, 'WS 2 TP 1' means this entity describes Workingstep 2, tool path 1.

The description attribute must have the value

```
'cutter location trajectory'
```

The other values allowed for this attribute in the full AP-238 standard are: `'axis trajectory', 'cutter contact trajectory', 'approach lift path angle', 'approach lift path tangent', 'connect security plane', 'connect direct'.`

The consequence attribute is not used in our CC1 subset. We recommend setting its value to the empty string.

The purpose attribute is not used in our CC1 subset. We recommend setting its value to the empty string.

ENTITY machining_toolpath_sequence_relationship

In the AP-238 CC1 subset a machining_toolpath_sequence_relationship entity is used to connect a machining_toolpath entity to a freeform_milling_operation entity so that the latter can describe a tool path for the former.

```
ENTITY machining_toolpath_sequence_relationship
  SUBTYPE OF (sequential_method);
      name              : STRING;
      description       : OPTIONAL STRING;
      relating_method   : action_method;
      related_method    : action_method;
      sequence_position : REAL;
END_ENTITY;
```

We recommend setting the value of the name attribute to the value of the name attribute of machining_toolpath entity.

We recommend setting the value of the description attribute to `'free form milling'`.

The relating_method attribute must contain the address of the freeform_milling_operation entity.

The related_method attribute must contain the address of the machining_toolpath entity.

The sequence_position attribute must contain the position of this machining_toolpath entity in the sequence of machining_toolpath entities belonging to the freeform_milling_operation entity. This value must be unique within the set of machining_toolpath_sequence_relationship entities that address a particular freeform_milling_operation entity. The value is defined as a real to allow for easier insertion of new toolpaths in an existing list.

ENTITY machining_workingstep

Machining_workingstep entities are used to describe workingsteps.

An AP-238 workingstep is defined by a feature and an operation. In the AP-238 CC1 OMAC subset features are not defined and the only operation supported is freeform_milling.

A machining workingstep entity is connected to its freeform_milling_operation entity by a machining_operation_relationship entity.

In the AP-238 CC1 OMAC subset there is one machining_workingstep for each freeform_milling_operation and vice versa.

An AP-238 machining workplan is made up of a sequence of working steps each of which perform an operation on the part. In the AP-238 CC1 subset there is only one workplan in the file and this workplan always consists of a sequence of unconditional workingsteps that are to be executed in sequence.

A machining_workingstep is connected to its workplan by a machining_process_sequence_relationship.

```
ENTITY machining_workingstep
      SUBTYPE OF (machining_process_executable);
      name        : STRING;
      description : STRING;
      consequence : STRING;
      purpose     : STRING;
END_ENTITY;
```

We recommend using the name attribute to describe the working step number. This name will be repeated in many of the entities that are connected to the workingstep. For example, 'WS 2' means this entity describes Workingstep 2.

We recommend setting the description attribute to a value meaningful to the end user and the empty string otherwise.

The consequence attribute is not used in our CC1 subset. We recommend setting its value to the empty string (").

The purpose attribute is not used in our CC1 subset. We recommend setting its value to the empty string (").

ENTITY machining_workplan

An AP-238 machining workplan is made up of a sequence of working steps each of which perform an operation on the part. In the AP-238 CC1 subset there is only one workplan in the file and this workplan always consists of a sequence of unconditional workingsteps that are to be executed in sequence.

In the AP-238 CC1 subset the workplan entity will always be linked to the machining_project entity that describes this machining project via a product_definition_process entity. In another AP-238 file this will only be the case if this workplan is the root of the project.

A machining_workplan is connected to its workingsteps by machining_process_sequence_relationship entities that will have sequence numbers.

```
ENTITY machining_workplan
      SUBTYPE OF (machining_process_executable);
      name        : STRING;
      description : STRING;
      consequence : STRING;
      purpose     : STRING;
END_ENTITY;
```

We recommend using the name attribute to describe the conformance class of this workplan. This name will be `'AP-238 CC1 OMAC Subset Version 1.0'` for workplans that conform to the specifications described in this document. If this value is the same as the value stored in the application attribute of the application_context entity then the workplan will conform to the same requirements as the rest of the AP-238 file. In the future, there may be examples of AP-238 CC1 OMAC Subset workplans stored in files that conform to another conformance class.

We recommend the description attribute be set to a value that is meaningful to the end user or the empty string (").

The consequence attribute is not used in our CC1 subset. We recommend setting its value to the empty string (").

The purpose attribute is not used in our CC1 subset. We recommend setting its value to the empty string (").

ENTITY measure_representation_item

A measure representation item is used to store a quantity. The name of the item is used to describe the type of the quantity, the value component is used to describe the value and the unit component is used to describe the unit. A measure_representation_item is addressed by a representation entity that will be linked to the rest of the data set by action_property entities.

```
ENTITY measure_representation_item
  SUBTYPE OF (representation_item, measure_with_unit);
      name   : STRING;
      value_component : measure_value;
      unit_component  : unit;
END_ENTITY;
```

In the AP-238 CC1 subset the name must be one of the following values.

```
'rotational speed',  'feed speed'
```

A rotational speed is used to describe the spindle speed. The following text describes an instance of a measure_representation_item for a rotational_speed.

```
#2373=MEASURE_REPRESENTATION_ITEM('rotational speed',NUMERIC_MEASURE(2000.), #5000);
```

In this text the string NUMERIC_MEASURE is necessary because there are many ways to measure quantities. The example shown is a number with the simple value 2000.0. The unit component attribute addresses an entity that describes units used to measure the quantity. In the AP-238 subset the units used to describe a rotational quantity are revolutions per minute. These units can be inserted into the Part21 file using the following lines of text.

```
#5000=DERIVED_UNIT((#5010));               /* rpm  (1/minute) */
#5010=DERIVED_UNIT_ELEMENT(#5020,-1.);
#5020=(                                   /* minute -> 60s */
CONVERSION_BASED_UNIT('MINUTE',#5030)
TIME_UNIT()
NAMED_UNIT(#5040)
);
#5030=TIME_MEASURE_WITH_UNIT(TIME_MEASURE(60.0),#5050);
#5040=DIMENSIONAL_EXPONENTS(0.,0.,1.,0.,0.,0.,0.);
#5050=(                                   /* second */
TIME_UNIT()
NAMED_UNIT(*)
```

```
SI_UNIT($,.SECOND.)
);
```

A feed speed is used to describe the cutting speed of the tool as it move along the tool path. The following text describes an instance of a measure_representation_item for a feed_speed.

```
#2374=MEASURE_REPRESENTATION_ITEM('feed speed',NUMERIC_MEASURE(500.),#6000);
```

The example shown has a feed speed of 500.0. The unit component addresses an entity that describes the unit of the feed. In the AP-238 CC1 subset feeds are measured in inches per minute. These units can be inserted into the Part 21 file using the following lines of text.

```
#6000=DERIVED_UNIT((#6010,#6020));          /* inch/minute */
#6010=DERIVED_UNIT_ELEMENT(#6030,1.);
#6020=DERIVED_UNIT_ELEMENT(#6060,-1.);
#6030=(                                     /* inch -> 25.4mm */
CONVERSION_BASED_UNIT('INCH',#6040)
LENGTH_UNIT()
NAMED_UNIT(#6050)
);
#6040=LENGTH_MEASURE_WITH_UNIT(LENGTH_MEASURE(25.4),#6090);
#6050=DIMENSIONAL_EXPONENTS(1.,0.,0.,0.,0.,0.,0.);
#6060=(                                     /* minute -> 60s */
CONVERSION_BASED_UNIT('MINUTE',#6070)
TIME_UNIT()
NAMED_UNIT(#6080)
);
#6070=TIME_MEASURE_WITH_UNIT(TIME_MEASURE(60.0),#6100);
#6080=DIMENSIONAL_EXPONENTS(0.,0.,1.,0.,0.,0.,0.);
#6090=(                                     /* millimeter */
LENGTH_UNIT()
NAMED_UNIT(*)
SI_UNIT(.MILLI.,.METRE.)
);
#6100=(                                     /* second */
TIME_UNIT()
NAMED_UNIT(*)
SI_UNIT($,.SECOND.)
);
```

ENTITY polyline

Polylines are used to hold the list of points in the basic curve that defines a machining_toolpath and the list of unit vectors in the tool axis of a 5-axis tool path.

Polylines are linked to their machining_toolpath entities by a representation entity that is attached to a action_property_representation entity that is attached to an action_property entity..

The Polylines that define the basic curve and tool axis for a 5-axis machining_toolpath must contain the same number of points.

```
ENTITY polyline
      SUBTYPE OF (bounded_curve);
      name   : STRING;
      points : LIST [2:?] OF cartesian_point;
END_ENTITY;
```

The AP-238 CC1 OMAC subset recommends setting the name of the Polyline to be the same value as name of the machining_toolpath _entity that contains this polyline.

The standard requires every polyline to contain at least two cartesian_points.

ENTITY process_product_association

A process product association entity is used to connect a machining_project to the workplan that defines the workingsteps in the project.

```
ENTITY process_product_association;
    name            : STRING;
    description     : STRING;
    defined_product : characterized_product_definition;
    process         : product_definition_process;
END_ENTITY;
```

The AP-238 OMAC CC1 subset recommends setting the value of the name attribute to the same value as the name attribute of the machining_project entity connected to this entity via the defined_product attribute.

The AP-238 OMAC CC1 subset recommends setting the value of the description attribute to the empty string.

The standard requires the defined_product attribute to be set to the address of the product_definition that represents the project.

The standard requires the process attribute be set to the address of the product_definition_process entity for the workplan that is root of the machining process.

ENTITY product
(for workpiece)

STEP defines data for many different kinds products. AP-203 defines generic mechanical product data. AP-238 defines CNC machining product data.

In AP-238 a product entity that represents a project will have the type machining_project which is a subtype of product.

If an entity instance has the type product then in the AP-238 CC1 subset it must represent the workpiece. In an AP-238 OMAC subset there can only be one product entity instance and it must represent the workpiece. In the AP-238 standard there can be many products representing the workpiece, the stock, the fixtures and the tools.

The product entity will be addressed by a product_definition_formation entity that is addressed by a product_definition entity. The product_definition_formation entity will describe the version number of this workpiece. In an AP-238 CC2 or above file the product_definition entity will lead to additional data about the workpiece such as the features being manufactured and the required tolerances of those features.

In the OMAC AP-238 CC1 subset there will only be one product_definition_formation entity for each product entity and there will only be one product_definition entity for each product_definition_formation entity.

```
ENTITY product
    id                : STRING;
    name              : STRING;
    description       : OPTIONAL STRING;
    frame_of_reference : SET [1:?] OF product_context;
END_ENTITY;
```

The OMAC AP-238 CC1 subset recommends setting the id attribute to the organizations identifier for this workpiece product.

The OMAC AP-238 CC1 subset recommends setting the name attribute to the organizations informal name for the workpiece product if there is one and the same value as the id attribute otherwise.

The OMAC AP-238 CC1 subset recommends setting the description attribute to end user data or the empty string.

The standard requires the frame_of_reference attribute to be set to the address of a product_context entity. If the workpiece product entity was defined by another application that set this context data to the appropriate value for the application protocol used by the application then this product_context entity should not be changed for this data exchange file. If the product_definition is new, however, then it should be set to the values described for the product_context entity in this reference.

ENTITY product_context

STEP defines data for many different kinds products. AP-203 defines generic mechanical product data. AP-238 defines CNC machining product data.

The product_context entity is addressed by a machining_project entity and defines context information for that entity.

The product_context entity may also be addressed by a product entity that represents the workpiece being made by the project. However, this addressing will only take place if the workpiece entity has been newly defined for this data set. If the workpiece product entity was defined in another context, for example, by a CAD application in the AP-203 design context then the product entity should continue to reference the context entity created by the AP-203 or other application.

The product_context entity is closely related to a product_definition_context entity that defines similar information that is specific to the definition of the product that defines this version of the product. There should only be one instance of each in an AP-238 CC1 OMAC subset file.

Both entities contain information for the end-user of the CNC program defined by the AP-238 CC1 file. A system that reads this information is not required to determine that the information given in these entities is consistent with the data described by the workingsteps.

In the AP-238 CC1 OMAC subset the main difference between these entities is that the name attribute of this entity describes a restriction on the range of operations in the file that will apply to all versions of the project while the name attribute of the product_definition_context entity describes a restriction for a specific this version of the file if that restriction is more restrictive than the one described here. For example, it may be the intent for this machining project to contain milling and drilling operations while the current version only contains milling operations.

```
ENTITY product_context
  SUBTYPE OF (application_context_element);
      name                    : STRING;
      frame_of_reference      : application_context;
      discipline_type         : STRING;
END_ENTITY;
```

The AP-238 CC1 OMAC subset recommends the name attribute is set to one of the following:

'drilling'	if the CNC program for this product will only contain drilling operations
'milling'	if the CNC program for this product will only contain milling operations
'turning'	if the CNC program for this product will only contain milling operations
'drilling and milling'	if the CNC program for this product will only contains drilling and milling operations
'drilling, milling and turning'	if the CNC program for this product will contain drilling, milling and turning operations
'drilling and turning'	if the CNC program for this product will only contain drilling and turning operations

`'milling and turning'`	if the CNC program for this product will only contain milling and turning operations

The standard requires the frame_of_reference attribute be set to the address an application_context entity.

The discipline attribute should be set to the value `'CNC machining'`.

ENTITY product_definition
(for machining_project)

A product definition entity is used to define a product. Many different kinds of products are allowed in a STEP file. The new type of product allowed by AP-238 is a machining_project product.

If a product_definition defines a project then the formation attribute will contain the address of a product definition formation entity that addresses a machining_project entity in its of_product attribute.

If a product_definition defines a project then there will be a process_product_association in the file that connects this product_definition entity to a machining_workplan entity via a product_definition_process entity.

If a product_definition defines a project there will be a machining_project_workpiece_relationship entity in the file that connects this product_definition entity to the product_definition entity that represents the project. This product_definition will be addressed by the machining_workpiece_relationship entity in its relating_product_definition attribute and the product_definition entity that represents the workpiece will be addressed by the machining_workpiece_relationship entity in its related_product_definition attribute.

The descriptions below assume this entity is a product_definition entity for a machining_project entity.

```
ENTITY product_definition;
    id                  : STRING;
    description         : STRING;
    formation           : product_definition_formation;
    frame_of_reference : product_definition_context;
END_ENTITY;
```

The AP-238 CC1 subset recommends setting the value of the id attribute to the same value as the machining_project entity name attribute.

The AP-238 CC1 subset recommends setting the value of the description attribute to a value meaningful to user or to the empty string.

The standard requires the formation attribute to address a product_definition_formation entity that addresses a machining_project entity in its of_product attribute.

The standard requires the frame_of_reference entity to address a product_definition_context entity that describes additional product definition information for the end_user.

ENTITY product_definition
(for workpiece)

A product definition entity is used to define a product. Many different kinds of products are allowed in a STEP file.

If a product_definition defines a workpiece then the formation attribute will contain the address of a product definition formation entity that addresses a product entity in its of_product attribute.

The workpiece is used to define the product being made by the CNC machining project.

For each product_definition that defines a workpiece there will be a machining_project_workpiece_relationship entity in the file that connects this product_definition entity to the product_definition entity that represents the project. This product_definition will be addressed by the machining_workpiece_relationship entity in its related_product_definition attribute and the project entity will be addressed by the machining_workpiece_relationship entity in its related_product_definition attribute.

In the AP-238 CC1 subset there is only one workpiece for each project. The AP-238 standard allows additional workpieces to define the stock and other products used in the set-up and fixture-ing.

The descriptions below assume this entity is a product_definition entity for the workpiece.

```
ENTITY product_definition;
    id                 : STRING;
    description        : STRING;
    formation          : product_definition_formation;
    frame_of_reference : product_definition_context;
END_ENTITY;
```

The AP-238 CC1 subset recommends setting the value of the id attribute to the same value as the name attribute of the product entity name connected to the product_definition entity via its formation attribute.

The AP-238 CC1 subset recommends setting the value of the description attribute to a value meaningful to user or to the empty string.

The standard requires the formation attribute to address a product_definition_formation entity that addresses a product entity in its of_product attribute.

The standard requires the frame_of_reference entity to address a product_definition_context entity that describes additional product definition information for the end_user. If the workpiece product was defined in the context of another Application Protocol then this information should be as defined for that protocol. If the workpiece product is being defined in the context of AP-238 then it should be defined as described for the product_definition_context entity in this document.

ENTITY product_definition_context

STEP defines data for many different kinds products at many different life cycle stages. AP-203 defines generic mechanical product data for the design life cycle stage. AP-238 defines CNC machining product data for the manufacturing life cycle stage.

The product_definition_conext entity is addressed by a product_definition entity and defines context information for that entity.

The product_definition_context entity may also be addressed by a product_definition entity that represents the workpiece being made by the project. However, this addressing will only take place if the workpiece entity has been newly defined for this data set. If the workpiece product_definition entity was defined in another context, for example, by a CAD application in the AP-203 design context then the product entity should continue to reference the context entity created by the AP-203 application.

Both entities contain information for the end-user of the CNC program defined by the AP-238 CC1 file. A system that reads this information is not required to determine that the information given in these entities is consistent with the data described by the workingsteps.

The product_definition_context entity is very similar to the product_context entity that will be linked to the product_definition entity via a product_definition_formation entity and a machining_project entity. The product_definition_context entity describes information that is specific to this version of the project while the product_context entity describes information that is going to be generic to all versions of the project. There should only be one instance of a product_definition_context entity in an AP-238 CC1 OMAC subset file.

```
ENTITY product_definition_context
  SUBTYPE OF (application_context_element);
      name                    : STRING;
      frame_of_reference      : application_context;
      life_cycle_stage        : STRING;
END_ENTITY;
```

In the OMAC AP-238 CC1 subset, if the range of operations in the current machining project is more specific than the range described in the product_context entity then the range should be described as follows:

'drilling'	if the CNC program for this product definition only contains drilling operations
'milling'	if the CNC program for this product definition only contains milling operations
'turning'	if the CNC program for this product definition only contains milling operations
'drilling and milling'	if the CNC program for this product definition only contains drilling and milling operations
'drilling, milling and turning'	if the CNC program for this product definition contains drilling, milling and turning operations
'drilling and turning'	if the CNC program for this product definition only contains drilling and turning operations

`'milling and turning'`	if the CNC program for this product definition only contains milling and turning operations

If the range is the same as that described in the product_context entity or the difference in range does not matter then the name attribute should be set to the empty string.

The frame_of_reference attribute should address an application_context entity that defines this file to be an AP-238 file.

The life_cycle_stage attribute should be set to the value `'manufacturing'`.

ENTITY product_definition_formation
(for machining_project)

STEP allows a product to have multiple versions and for each version to have multiple definitions. Each version is represented by a product_definition_formation entity.

In the OMAC AP-238 CC1 subset there will only be one product_definition_formation entity for each machining_project entity and there will only be one product_definition entity for each product_definition_formation entity.

If the end-user project that is the source of the project data in an AP-238 file is being versioned, then the id attribute of the product_definition_formation entity for the project should describe this version number.

The following descriptions assume this product_definition_formation entity describes a project.

```
ENTITY product_definition_formation;
    id          : STRING;
    description : OPTIONAL STRING;
    of_product  : product;
END_ENTITY;
```

The OMAC CC1 subset recommends setting the id attribute to the empty string if the project is not being versioned, and the version identifier otherwise.

The OMAC CC1 subset recommends setting the description attribute to NULL or a value that is meaningful to the end user if one has been supplied.

The standard requires the of_product be set to the address of the project entity that represents this machining project.

ENTITY product_definition_formation
(for workpiece)

STEP allows a product to have multiple versions and for each version to have multiple definitions. Each version is represented by a product_definition_formation entity.

In the OMAC AP-238 CC1 subset there will only be one product_definition_formation entity for each product entity and there will only be one product_definition entity for each product_definition_formation entity.

If the end-user project that is the source of the workpiece data in an AP-238 file is being versioned, then the id attribute of the product_definition_formation entity for the workpiece should describe this version number.

The following descriptions assume this product_definition_formation entity describes a workpiece.

```
ENTITY product_definition_formation;
    id          : STRING;
    description : OPTIONAL STRING;
    of_product  : product;
END_ENTITY;
```

The OMAC CC1 subset recommends setting the id attribute to the empty string if the project is not being versioned, and the version identifier otherwise.

The OMAC CC1 subset recommends setting the description attribute to NULL or a value that is meaningful to the end user if one has been supplied.

The standard requires the of_product be set to the address of the product entity that represents this workpiece.

ENTITY product_definition_process

A product definition process entity is used to show that a workplan is the root of a production process. In the AP-238 CC1 OMAC subset there is only one workplan but in a full AP-238 file there can be many workplans.

```
ENTITY product_definition_process
  SUBTYPE OF (action);
      name               : STRING;
      description        : OPTIONAL STRING;
      chosen_method      : action_method;
      identification     : STRING;
END_ENTITY;
```

The standard requires the name attribute to contain the value `'machining'`.

We suggest the description attribute is set to some value that is meaningful to the user. If no value has been supplied then it should be set to NULL.

The standard requires the chosen_method attribute be set to the address of the machining_workplan entity that is the root of the machining process.

We suggest the identification attribute is set to an identification value that is meaningful to the user. If no value has been supplied then it should be set to the empty string (`''`).

ENTITY representation

Every property value has a representation. A representation entity is used to hold all of the items necessary to describe the representation.

There are three types of representations: representations for constant values that can be shared by any workingstep and toolpath; representations for specific values that need to be set for each workingstep and toolpath; and type specific representations.

The type specific representation subtypes are called machining_feed_speed_representation and machining_spindle_speed representation respectively and they are described elsewhere in this document.

Representations can be shared by multiple workingsteps and toolpaths. For example, there only need to be one representation for the value ‘coolant on’ in the file and one representation for the value ‘coolant off’.

```
ENTITY representation;
    name             : STRING;
    items            : SET [1:?] OF representation_item;
    context_of_items : representation_context;
END_ENTITY;
```

The AP-238 CC1 OMAC subset recommend using the name to describe the type of value being represented.

`'basic curve'`	to indicate that this is the representation of the x, y, z coordinates of a tool path
`'tool axis'`	to indicate that this is the representation of the i, j, k, coordinates of a tool axis
`'coolant'`	to indicate that this is the representation of a coolant value
`'required constant'`	to indicate that this is the representation of a constant value `'required'`
`'tcp constant'`	to indicate that this is the representation of a constant value `'tool center point'`

The items attribute is a pointer to one or more values describing the representation.

If the representation is a spindle speed then the items attribute will address a single measure_representation_item entity.
If the representation is a basic curve then the items attribute will address a single polyline entity.
If the representation is a tool axis then the items attribute will address a single polyline entity.
If the representation is a feed speed then the items attribute will address a single measure_representation_item entity.

If the representation is the coolant value then the items attribute will address one of the following entities:

#9900=DESCRIPTIVE_REPRESENTATION_ITEM('shared constant','coolant on');
#9901=DESCRIPTIVE_REPRESENTATION_ITEM('shared constant','coolant off');

If the representation is the required constant value then the items attribute will address the following entity
#9902=DESCRIPTIVE_REPRESENTATION_ITEM('shared constant','required');

If the representation is the shared tcp constant value then the items attribute will address the following entity
#9903=DESCRIPTIVE_REPRESENTATION_ITEM('shared constant','tool center point');

ENTITY representation_context

A representation_context entity describes the global properties of all the items in the representation. Examples of the kinds of properties that may need to be described include the units and accuracy of the geometry in an AP-203 file.

In the AP-238 CC1 subset two context instances need to be described.

```
ENTITY representation_context;
    context_identifier : STRING;
    context_type       : STRING;
END_ENTITY;
```

The first kind of context is used for descriptive items where no units are necessary because the items are self-describing.. This context is used for the representation of tool axis, the required constant, and the tcp constant. The following entity instance can be added to the AP-238 Part 21 file for these units.

```
#8000=REPRESENTATION_CONTEXT('','units NOT necessary');
```

The second kind of context is used for the representation of the units of a polyline. In the AP-238 CC1 OMAC subset these units are always measured in inches. The following lines of code can be added to the AP-238 Part 21 file to describe these units. The EXPRESS definition of these units uses a complex form of inheritance to add additional data to the basic representation_context data. The lines of code shown define the length measure to be in inches, the angle measure to be in degrees and the cone measure to be in steridians.

```
#9800=(GEOMETRIC_REPRESENTATION_CONTEXT(3)
GLOBAL_UNIT_ASSIGNED_CONTEXT((#9700,#9600,#9500))
REPRESENTATION_CONTEXT(' ',' ')) ;
```

The following four entities must be in the file to define inches.

```
#9701=(
   CONVERSION_BASED_UNIT('INCH',#9701)
   LENGTH_UNIT()
   NAMED_UNIT(#9703)
);
#9701=LENGTH_MEASURE_WITH_UNIT(LENGTH_MEASURE(25.4),#9702);
#9702=(
   LENGTH_UNIT()
```

```
   NAMED_UNIT(*)
   SI_UNIT(.MILLI.,.METRE.)
);
#9703=DIMENSIONAL_EXPONENTS(1.,0.,0.,0.,0.,0.,0.);
```

The following four entities must be in the file to define degrees.

```
#9600=(
   CONVERSION_BASED_UNIT('DEGREE',#9601)
   NAMED_UNIT(#9603)
   PLANE_ANGLE_UNIT()
);
#9601=PLANE_ANGLE_MEASURE_WITH_UNIT(PLANE_ANGLE_MEASURE(0.01745329251994),
        #9602);
#9602=(
   NAMED_UNIT(*)
   PLANE_ANGLE_UNIT()
   SI_UNIT($,.RADIAN.)
);
#9603=DIMENSIONAL_EXPONENTS(0.,0.,0.,0.,0.,0.,0.);
```

The following entity defines the unit of measure for sterdians. It is a way to measure cones and other volume based angles. This unit is simpler because it is metric. The imperial equivalent is “barns”.

```
#9500=(
   NAMED_UNIT(*)
   SI_UNIT($,.STERADIAN.)
   SOLID_ANGLE_UNIT()
);
```

Appendix A
How to Read a STEP Part 21 File

Purpose

This Appendix describes the syntax of a STEP Part 21 data exchange file. It is a quick introduction to the major features. The full specification is given in ISO 10303-21.

History

The STEP Part 21 format is an ASCII data format that was invented as a means for exchanging data defined by an EXPRESS model by the STEP community. The format was developed several years before XML became popular as a way to exchange data between systems. It was invented in an era when file size was an important consideration, and relational databases were the leading influence on the development of data formats. As a consequence Part 21 is a non-redundant data format that does not include any details that can be deduced from information given elsewhere in the file.

The XML format has shown that including redundant data in a format makes the data much easier for programmers and application experts to understand and that the increased productivity of these experts is more important than the wasted data space. STEP has been seeking to replace Part 21 with an XML format known as Part 28. Unfortunately the replacement has not been easy because everyone has different ideas on what aspect of STEP needs to be improved by the new format. Therefore, Part 28 XML formats have been suggested that make the inheritance model easier to understand, the EXPRESS model easier to understand and the ARM model easier to understand. As yet there has been no consensus on the best solution and all of the solutions suggested (except ours) make something else in the STEP standards harder to understand.

The AP-238 standard includes an extensive appendix of XML examples for most of the data defined by AP-238 and the authors of AP-238 are trying to make this format the STEP XML Part 28 standard.

STEP Tools Kits

Most people would not try to read and write XML data without using an XML tool kit such as Microsoft DOM. Even in the simplest format there are lots of special cases and “fringe” rules that are difficult to remember and much easier to handle if they are embedded into the underlying software.

The same is true for STEP Part 21 file data. Although most of the essential rules for processing this data are described here and you can quickly write a piece of software to process this data you probably should not because if the software is going to process all of the special cases it is going to become a significant piece of work. A much better approach is to purchase a toolkit that has already been debugged for all these special cases such as the **ST-Developer** tool kit offered by STEP Tools, Inc. This tool kit contains a library that will read and write Part 21 data defined by a Part 21 model. STEP Tools, Inc has developed C++ code and Java code samples showing how an application can create all of the entities defined in the AP-238 CC1 OMAC subset.

The Format

A Part 21 file is divided into a header section and a data section.

The header section contains administrative data about who wrote the data exchange file and what software was used to create and process the data. The header section also contains the name of the EXPRESS schema that defines the file. In an AP-238 file the name of the schema is 'INTEGRATED_CNC_SCHEMA' and this value must be stored in the header of the files in an object called the FILE_SCHEMA as shown below.

```
FILE_SCHEMA (('INTEGRATED_CNC_SCHEMA'));
```

If there is no FILE_SCHEMA object in the header then it is not a legal Part 21 file. If the FILE_SCHEMA object contains the name of a different schema then this Part 21 file is defined by one of the other STEP standards. For example if the object contains the string 'CONFIG_CONTROL_DESIGN' then the file is defined by AP-203.

The data section contains the entities in the file. The data section begins with the keyword DATA; and ends with the keyword ENDSEC.

The data section describes the entity instances in the file. Each instance is described once only. Each description begins with an identifier and terminates with a semi-colon. Most entity instances are written on a single line of text but some instances span multiple lines because carriage returns can be include in the data wherever white space is allowed.

The identifier is an integer preceded by the “#” character. Every entity in the file must have a unique identifier. Usually, but not always the entity instances will be written in ascending order. However, this is not required and an application that reads the file is not required to preserve the identifiers so it may read in the file and then write out the same data with a completely different set of identifiers.

The identifier is followed by an “=” symbol. For most entity instances this identifier is then followed by the name of the EXPRESS definition of the entity. The name is always written in UPPER case because EXPRESS is a case insensitive language.

This name is then followed by a comma-separated list of the values assigned to the attributes of the entity. The comma-separated list is enclosed in parentheses. The syntax of each value depends on the type of the attribute. The following line of code describes a value for an instance of an action_property_representation entity.

```
#1297=ACTION_PROPERTY_REPRESENTATION('basic curve','',#1898,#738);
```

The EXPRESS model for this entity is as shown below.

```
ENTITY action_property_representation;
        name           : STRING;
        description    : STRING;
        property       : action_property;
        representation : representation;
END_ENTITY;
```

The EXPRESS model shows the entity has four attributes. Therefore, the line of code in the Part 21 file defines four values for the instance. The order is significant. The first value in the Part 21 defines the first attribute of the EXPRESS and so on. Therefore, in this instance the value of the name attribute is “basic curve”, the value of the description attribute is the empty string “”, the value of the property attribute is the entity instance with the identifier “#1898” and the value of the representation attribute is the entity instance with the identifier “#738”.

If you look up the entity instance referenced by an attribute it will have the type defined by the type of the attribute in the EXPRESS, but EXPRESS has a very powerful inheritance model and in many cases the instance at the other end of the reference will belong to one of the sub-types of the attribute type. In AP-238, the action_property entity is not sub-typed so it will have the type shown. However, the representation entity is sub-typed so the instance could be a machine_feed_speed_representation or a machine_spindle_speed_representation depending on the context.

Each of the primitive types allowed in EXPRESS has its own signature in a Part 21 file

STRING	Value will begin and end with single quote values. The normal back slash convention is used for embedded quotes. There are elaborate rules for non-standard characters but they are not used much in practice
INTEGER	The value will contain digits only e.g “10” or “2991” or “0”
REAL	The value will begin with a digit and contain a decimal point e.g “10.3” or “10.” or “0.”. An exponent is also allowed (see the standard for the rules).
ENUMERATION	The value will begin an end with a period e,g “.RED.”
BOOLEAN	The value will be written as an enumeration with the value “.T.” or “.F.”.

A NULL value is shown as a dollar symbol in the file "$". In a fully conformant Part 21 file only OPTIONAL attributes are allowed to be NULL. Therefore, empty strings are frequently used to define string attributes because there is no better value to populate the data.

A list value is shown by nested parentheses. These parentheses will be shown even if there is only one item in the list. If there is more than one item they will be comma separated. For example, the following line of code defines an instance of a representation.

```
#904=REPRESENTATION($,(#2458),$);
```

The EXPRESS definition for a representation is as follows

```
ENTITY representation
      SUBTYPE_OF (representation);
    name             : STRING;
    items            : SET [1:?] OF representation_item;
    context_of_items : representation_context;
END_ENTITY;
```

The items attribute is defined by a set. A set is stored in the file as a list that is required to contain unique values.

Note that the name attribute of the representation and the context_of_items attribute is defined to be OPTIONAL so they are not allowed to be NULL yet in the example they have NULL values. This is a fairly frequent occurrence in Part 21 files that have not been fully debugged. A tool kit will read and write this kind of data. A conformance checking tool will check this type of constraint and complain about it and your files will not be considered compliant until they obey all these secondary constraints. Another example of a secondary constraint is checking that the items in a SET are really unique.

Additional Features

We have now described the major features of a STEP Part 21 file. In addition to these major issues there are three subsidiary features that will be described briefly.

The first subsidiary issue is the treatment of a special kind of entity instance called an AND/OR instance. These AND/OR instances are defined by more than one type. They occur in the STEP models when a simple value from one perspective needs to be amplified by additional data to support another perspective. The treatment of tolerances is one example. Most models do not consider tolerances because they make the modeling too complicated. However, in a real implementation it needs to be possible to add a tolerance value (+/- etc) to most quantities. STEP handles this situation by allowing additional types to be added to the original quantity using a novel form of inheritance called AND/OR inheritance.

We do not need to describe AND/OR inheritance here but its impact on the Part 21 file needs to be described because if an entity instance is an AND/OR instance then it is not just described by one type so there is NO type name to list immediately after the "=" symbol in the Part 21 file. Instead all of the types of the entity are listed. This includes all of the inherited types. Each type then gives the values of the attributes defined in that type only. The following line of code shows an example:

```
#9602=(
   NAMED_UNIT(*)
   PLANE_ANGLE_UNIT()
   SI_UNIT($,.RADIAN.)
);
```

In this example instance 9602 has three types: named_unit, plane_angle_unit and si_unit. Notice that the three types are enclosed by parentheses but the list is not comma separated. In the example each type is shown on its own line in the file but this is not required. If you look at the EXPRESS definition of AP-238 you will see that named_unit is a supertype of plane_angle_unit. In a normal instance it would not be shown but as previously mentioned in the modified format all of the types are listed. In the Part 21 standard the modified format is called an external mapping.

The previous example also shows our second unusual feature. This is the "*" symbol in the name_unit entity. As previously mentioned the order of the attributes in an instance in a Part 21 file is significant. Therefore, if a subtype of an EXPRESS entity redefines a super-type attribute so that it does not exist in the sub-type then it cannot be deleted in the super-type because parsers will become confused in some cases. The star symbol "*" is used to show that the attribute has been redefined by a subtype so that the parsers will not become confused.

Finally, the last case we need to consider is the occasional use of names in the parameter list for an entity. Sometimes the type of an attribute is so complex that additional data is necessary for a parser to decode that type. In these rare cases the type is included in the value of the attribute as a name that contains the data of the attribute in parentheses. Measure representation item provides an example. In this entity the value "2000." is not sufficient to deduce that the attribute value is a NUMERIC_MEASURE because another type of value allowed for this attribute is also a REAL value. The distinction is resolved by enclosing the value with NUMERIC_MEASURE label.

```
#2373=MEASURE_REPRESENTATION_ITEM('rotational speed',NUMERIC_MEASURE(2000.), #5000);
```

Postscript

Unless your approach to reading a technical document is to only read the first and last paragraph, you will have noticed that many of the relationships in the AP-238 model are in the wrong direction from the perspective of an application that is trying to navigate the data from the root project entity down to the Cartesian point entities in a toolpath.

This is a deliberate feature of the STEP protocols because no one knows all of the attributes that are going to be necessary to describe all of the properties of a product across its complete life cycle. As new Application Protocols are being developed and as the old Application Protocols are being revised new properties are added to the model.

Because of the style used to create the STEP models the addition of the properties does not impact exiting applications because they were not looking for that data anyway.

The alternative approach is to put all of the properties into the entity as direct attributes. This is a better approach only while the model is perfect from all perspectives and only if all implementations agree to use all of the properties. In practice, depending on the perspective applications will have to leave some of the attributes as NULL, and there will be crucial attributes that are not in the model and they will have to be added to it using the STEP approach. I.E. by creating a new entity that references the old entity and the new property. Therefore, an application will have to learn how to navigate the data in the object style and the STEP style and the style chosen for each property will be random in the sense that it will depend on historical accident of which perspective (Application Protocol) defined the data first.

The STEP style is at least consistent in its awfulness. When navigating data in a Part 21 file you must always be aware that the application specific properties are attached to the data via reverse pointers. For example, the machining properties of a machining_toolpath entity are attached to that entity by action_property entities that address the machining_toolpath in one of their attributes and not vice versa.

A good STEP Toolkit will include a Part 21 file browser containing functionality that makes it easy for you to find and navigate these reverse relationships.